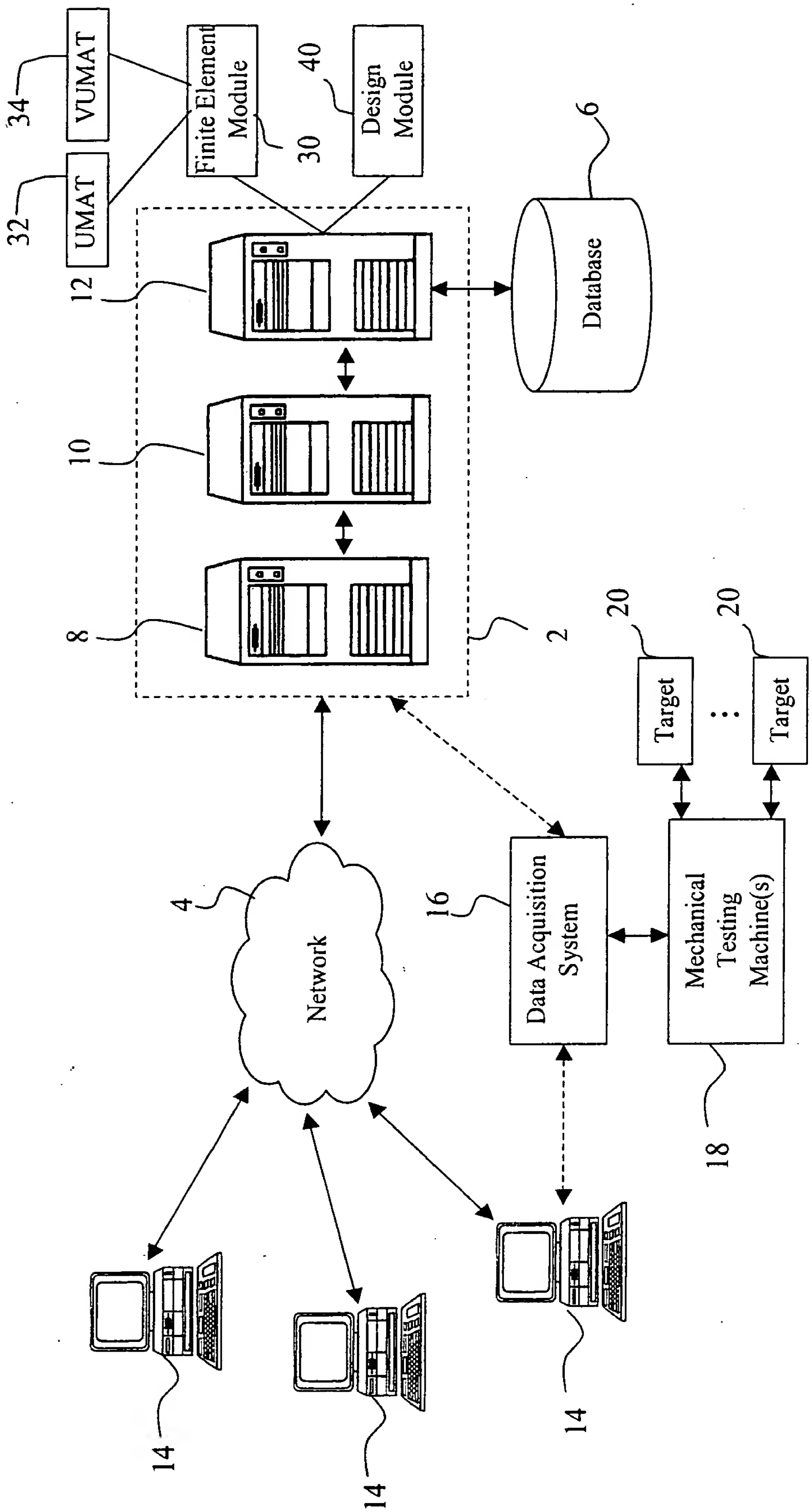
34
VUMAT
32
UMAT
Finite Element Module
30
40
Design Module
6
12
10
8
2
Database
20
Target
20
Target
4
Network
16
Data Acquisition System
Mechanical Testing Machine(s)
18
14
14
14

FIG. 1

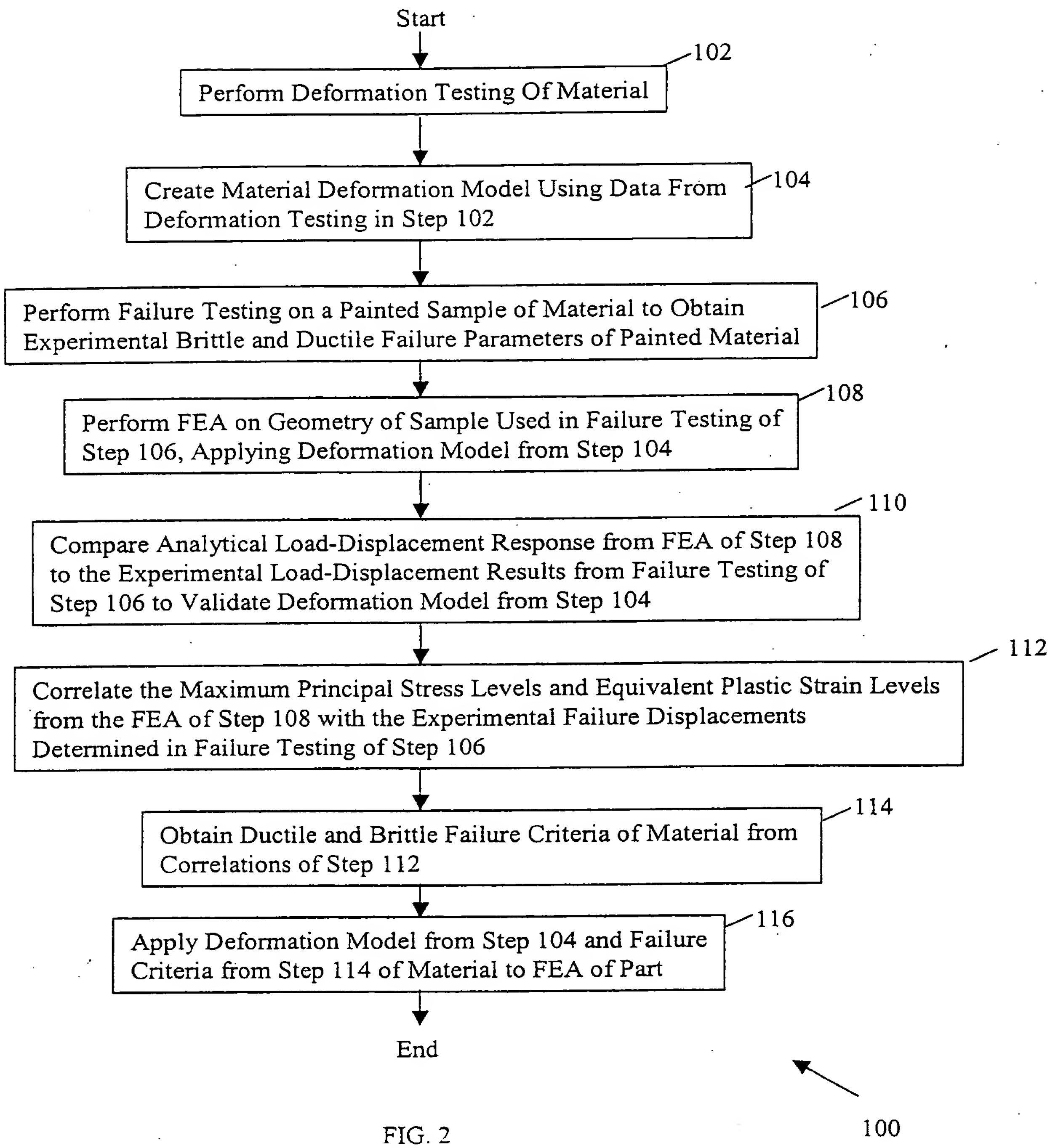
Start
Perform Deformation Testing Of Material
102
Create Material Deformation Model Using Data From Deformation Testing in Step 102
104
Perform Failure Testing on a Painted Sample of Material to Obtain Experimental Brittle and Ductile Failure Parameters of Painted Material
106
Perform FEA on Geometry of Sample Used in Failure Testing of Step 106, Applying Deformation Model from Step 104
108
Compare Analytical Load-Displacement Response from FEA of Step 108 to the Experimental Load-Displacement Results from Failure Testing of Step 106 to Validate Deformation Model from Step 104
110
Correlate the Maximum Principal Stress Levels and Equivalent Plastic Strain Levels from the FEA of Step 108 with the Experimental Failure Displacements Determined in Failure Testing of Step 106
112
Obtain Ductile and Brittle Failure Criteria of Material from Correlations of Step 112
114
Apply Deformation Model from Step 104 and Failure Criteria from Step 114 of Material to FEA of Part
116
End
100

FIG. 2

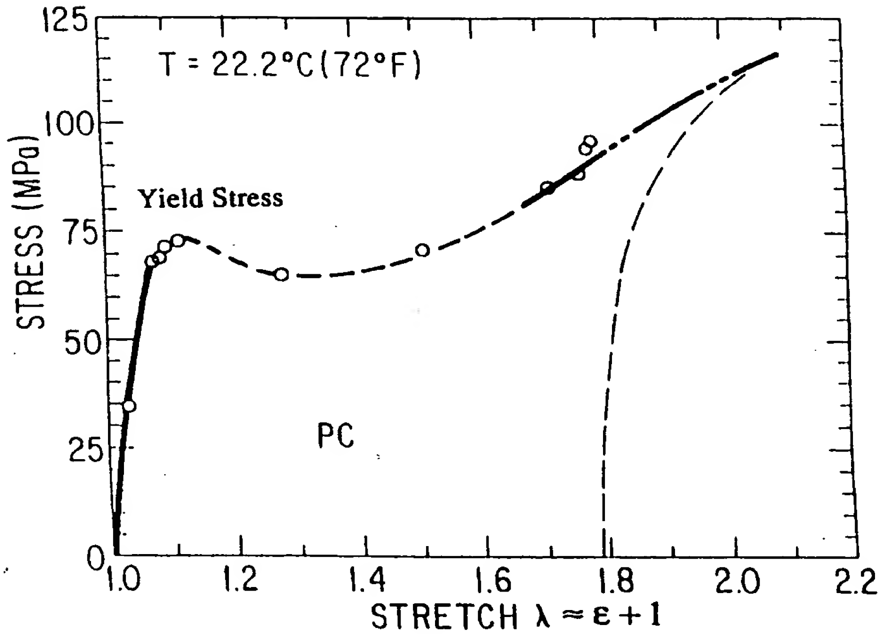
T = 22.2°C(72°F)
Yield Stress
PC
STRESS (MPa)
STRETCH λ ≈ ε + 1
125
100
75
50
25
0
1.0
1.2
1.4
1.6
1.8
2.0
2.2

FIG. 3

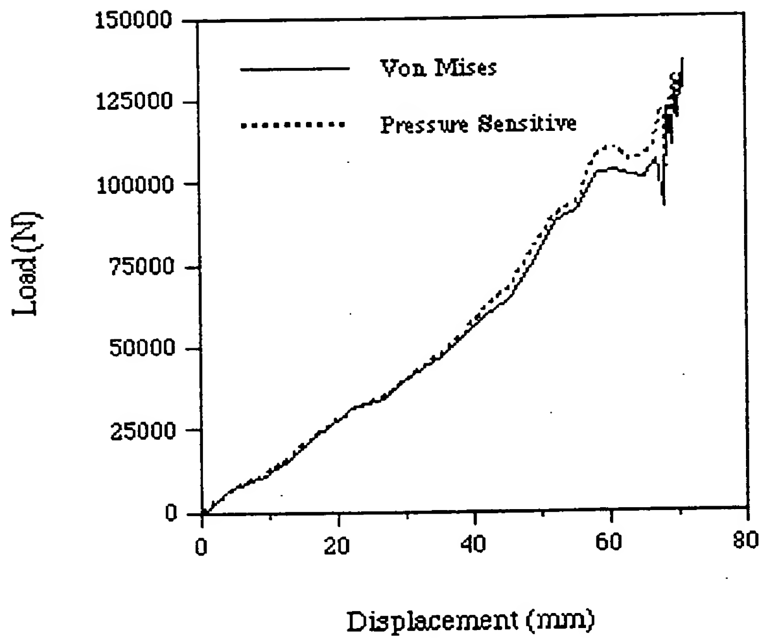
150000
125000
100000
75000
50000
25000
0
Load (N)
Von Mises
Pressure Sensitive
0
20
40
60
80
Displacement (mm)

FIG. 4

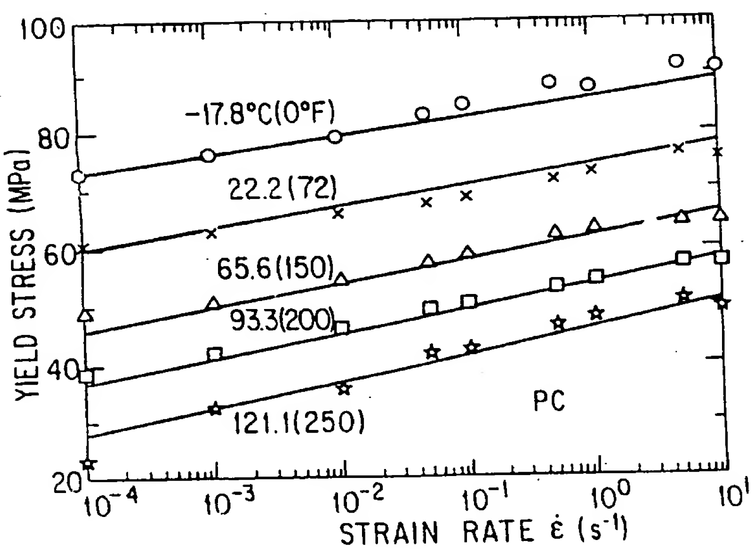
100
80
60
40
20
YIELD STRESS (MPa)
-17.8°C(0°F)
22.2 (72)
65.6 (150)
93.3 (200)
121.1 (250)
PC
10^{-4}
10^{-3}
10^{-2}
10^{-1}
10^{0}
10^{1}
STRAIN RATE $\dot{\varepsilon}$ (s^{-1})

FIG. 5

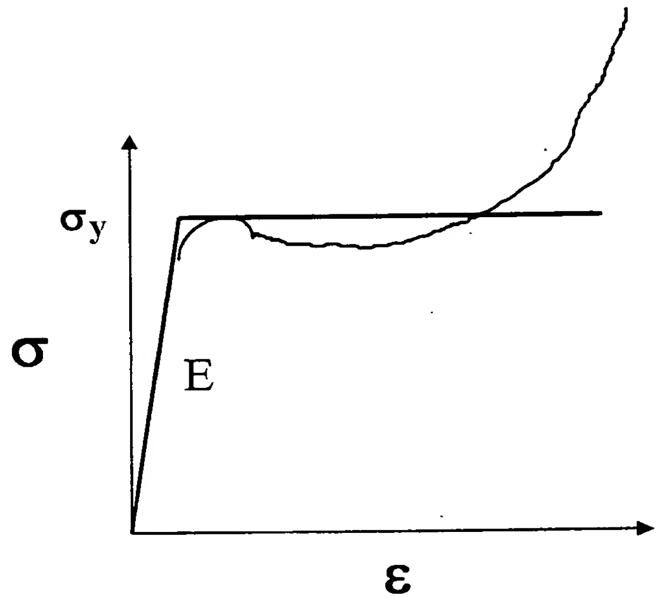

FIG. 6

Cantor Cc n LLP 55 Griffin Road South, Bloomfield, CT 0600
(860) 286-2929

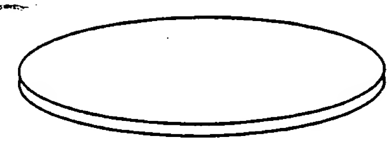

FIG. 7

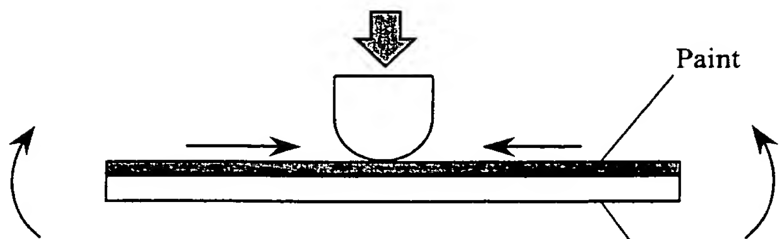

FIG. 8

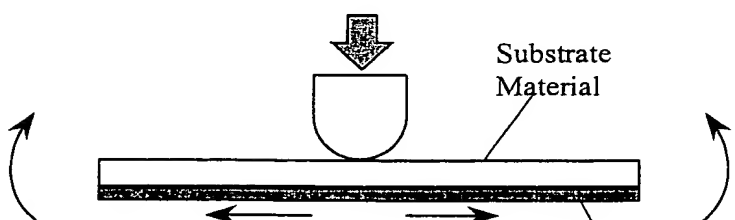

FIG. 9

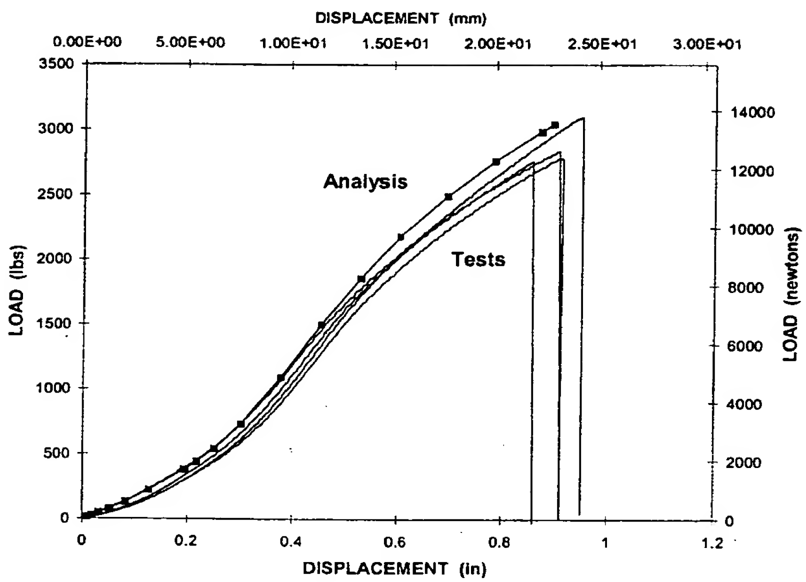
DISPLACEMENT (mm)
0.00E+00
5.00E+00
1.00E+01
1.50E+01
2.00E+01
2.50E+01
3.00E+01
Analysis
Tests
LOAD (lbs)
LOAD (newtons)
DISPLACEMENT (in)
0
0.2
0.4
0.6
0.8
1
1.2
500
1000
1500
2000
2500
3000
3500
2000
4000
6000
8000
10000
12000
14000

FIG. 10

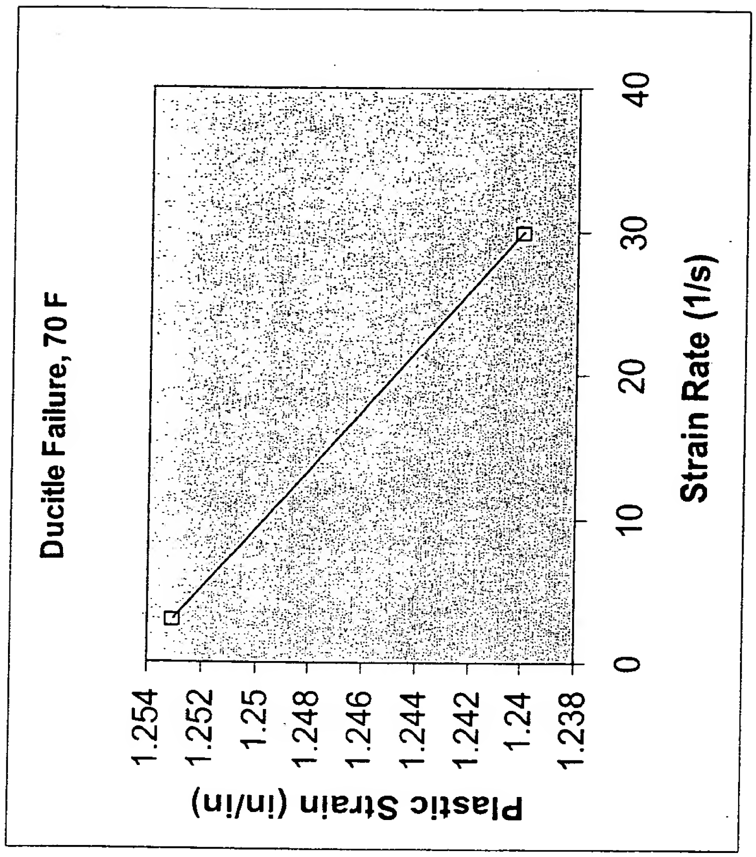
Ducitle Failure, 70 F
Plastic Strain (in/in)
1.254
1.252
1.25
1.248
1.246
1.244
1.242
1.24
1.238
0
10
20
30
40
Strain Rate (1/s)

FIG. 11

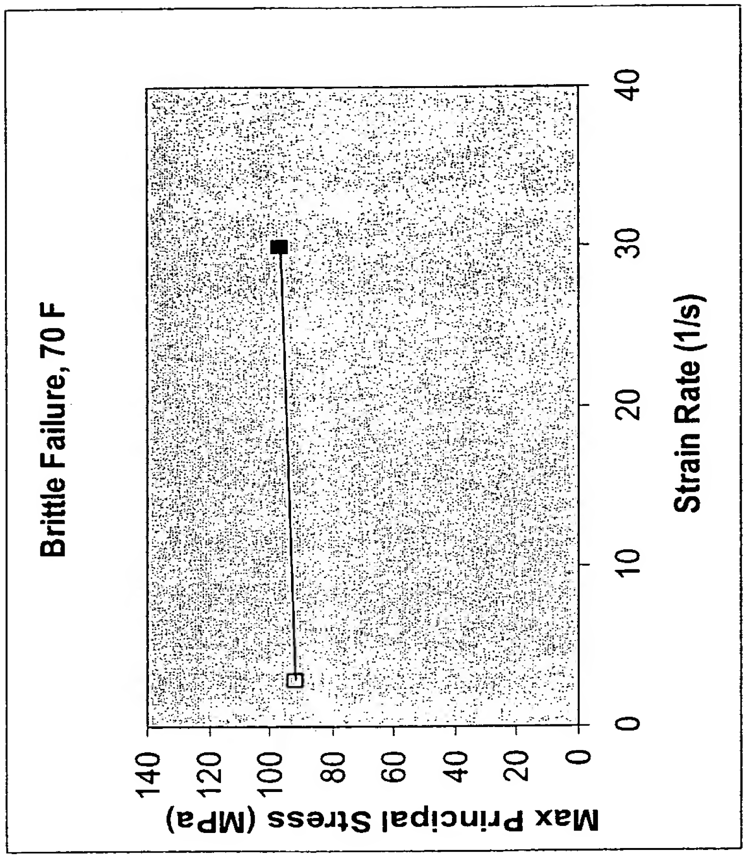
Brittle Failure, 70 F
Max Principal Stress (MPa)
140
120
100
80
60
40
20
0
0
10
20
30
40
Strain Rate (1/s)

FIG. 12

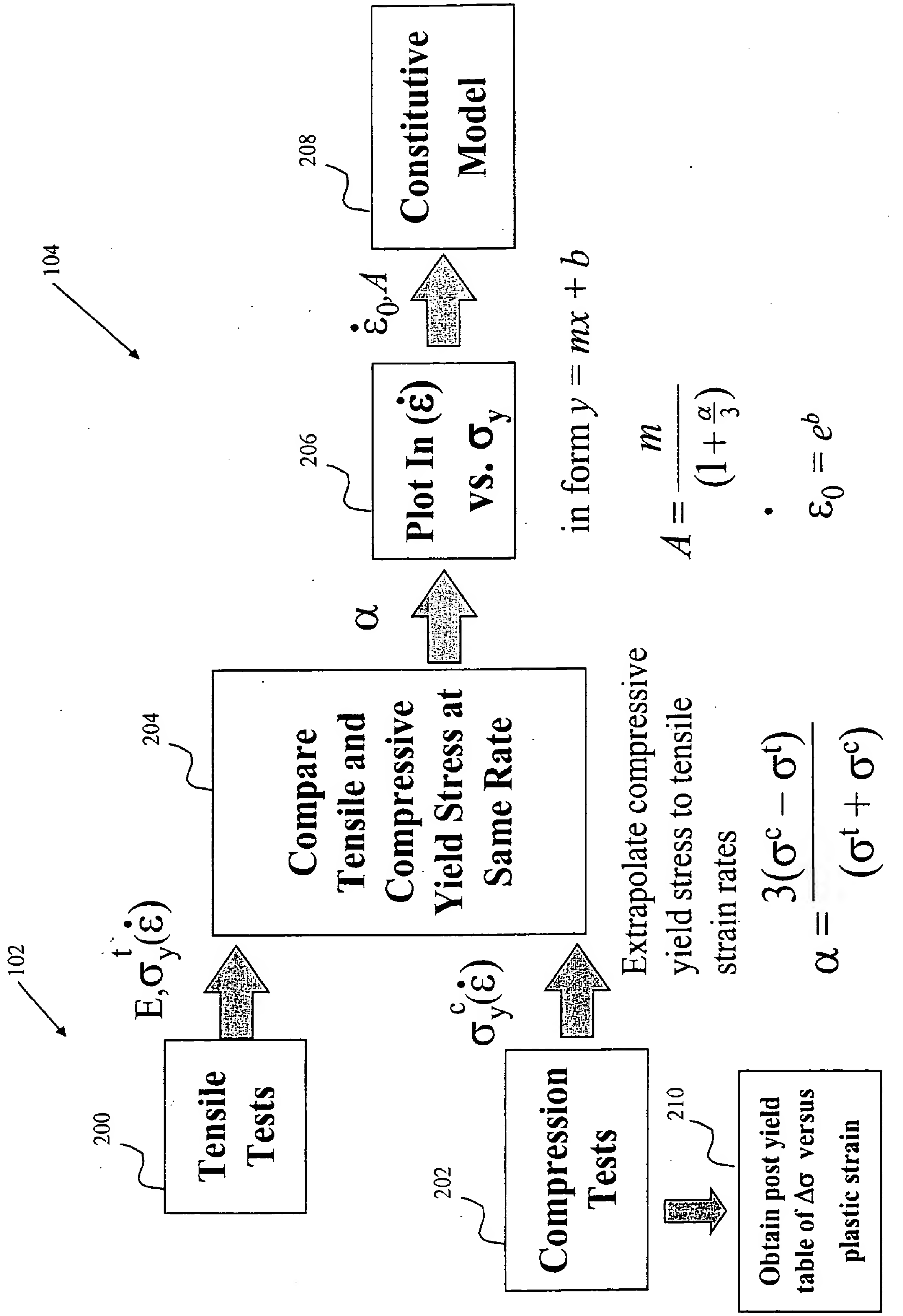
102
104
200
Tensile Tests
$E, \sigma_y^t(\dot{\varepsilon})$
202
Compression Tests
$\sigma_y^c(\dot{\varepsilon})$
Extrapolate compressive yield stress to tensile strain rates
$\alpha = \frac{3(\sigma^c - \sigma^t)}{(\sigma^t + \sigma^c)}$
210
Obtain post yield table of $\Delta\sigma$ versus plastic strain
204
Compare Tensile and Compressive Yield Stress at Same Rate
α
206
Plot In $(\dot{\varepsilon})$ vs. σ_y
in form $y = mx + b$
$A = \frac{m}{(1 + \frac{\alpha}{3})}$
$\dot{\varepsilon}_0 = e^b$
$\dot{\varepsilon}_0, A$
208
Constitutive Model

FIG. 13

source code implicit finite element solver

```
c       rate/temp/press dependent, von mises isotropic plasticity
c       umat for abaqus 5.5. nonlinear strain hardening.
c       2d/3d problems with the exception of plane stress
c       by omar a hasan        last modified 05-02-96
c       user must specify differential hardening data in umat
c       and dimension hardening table appropriately
c       must have atleast two sets of points in table
        subroutine umat(stress,statev,ddsdde,sse,spd,scd,
     1  rpl,ddsddt,drplde,drpldt,
     2  stran,dstran,time,dtime,temp,dtemp,predef,dpred,cmname,
     3  ndi,nshr,ntens,nstatv,props,nprops,coords,drot,pnewdt,
     4  celent,dfgrd0,dfgrd1,noel,npt,layer,kspt,kstep,kinc)
c
c
        include 'aba_param.inc'
c
        character*8 cmname
        dimension stress(ntens),statev(nstatv),
     1  ddsdde(ntens,ntens),ddsddt(ntens),drplde(ntens),
     2  stran(ntens),dstran(ntens),time(2),predef(1),pred(1),
     3  props(nprops),coords(3),drot(3,3),dfgrd0(3,3),dfgrd1(3,3)
c
        dimension flow(6)
c
        parameter(zero=0.d0,one=1.d0,two=2.d0,three=3.d0,six=6.d0,
     1  newton=60,toler=1.0d-5,twbth=0.66666666666d0)
c
c ------------------------------------------------------------------
c    cannot be used for plane stress
c ------------------------------------------------------------------
c    props(1) - e (Pa) (temperature dependent)
c    props(2) - nu
c    props(3) - rate sensitivity (temperature dependent)
c    props(4) - intrinsic flow rate (temperature dependent)
c    props(5) - pressure sensitivity
c    calls uhard for curve of intrinsic strength vs. plastic strain
c ------------------------------------------------------------------
c
c    material properties
      emod=props(1)
      enu=props(2)
      ebulk3=emod/(one-two*enu)
      eg2=emod/(one+enu)
      eg=eg2/two
      eg3=three*eg
      elam=(ebulk3-eg2)/three
      rlp2m=elam+eg2/three
      ratesf=props(3)
      rrates=one/ratesf
      dtebs0=dtime*props(4)
```

FIG. 14A

Cantor Colburn LLP 55 Griffin Road South, Bloomfield, CT 06002
(860) 286-2929

source code implicit finite element solver

```
      psf=props(5)
      esi=dstran(1)**2+dstran(2)**2+dstran(3)**2
      do k1=ndi+1,ntens
       esi=esi+two*(dstran(k1)/two)**2
      end do
      esi=sqrt(twbth*esi)
      s_rate=max(1.d-10,esi/dtime)
c
c     elastic stiffness
      call aset(ddsdde,zero,ntens*ntens)
      do k1=1,ndi
        do k2=1,ndi
          ddsdde(k2,k1)=elam
        end do
        ddsdde(k1,k1)=eg2+elam
      end do
      do k1=ndi+1,ntens
        ddsdde(k1,k1)=eg
      end do
c
c     recover equivalent plastic strain & equivalent stress
c     and hydrostatic stress at start of step
      eqplas=statev(1)
      qold=statev(2)
      hydr_o=(stress(1)+stress(2)+stress(3))/three
c
c     calculate predictor stress
      do k1=1,ntens
        do k2=1,ntens
          stress(k2)=stress(k2)+ddsdde(k2,k1)*dstran(k1)
        end do
      end do
c
c     calculate equivalent von mises stress
c
      smises=(stress(1)-stress(2))**2+(stress(2)-stress(3))**2
     1                                +(stress(3)-stress(1))**2
      do k1=ndi+1,ntens
        smises=smises+six*stress(k1)**2
      end do
      smises=sqrt(smises/two)
c
c     get differential hardening from the specified hardening curve
      call uhard(syiel0,hard,eqplas)
c
c     determine if actively yielding
      if (time(1).gt.0.d0) then
c
c       separate the hydrostatic from the deviatoric stress
c       calculate the flow direction
```

FIG 14B

Cantor Colburn LLP 55 Griffin Road South, Bloomfield, CT 06002
(860) 286-2929

source code implicit finite element solver

```
        shydro=(stress(1)+stress(2)+stress(3))/three
        do k1=1,ndi
          flow(k1)=(stress(k1)-shydro)/smises
        end do
        do k1=ndi+1,ntens
          flow(k1)=stress(k1)/smises
        end do
c
c      solve for equivalent von mises stress
c      and equivalent plastic strain increment using newton iterati
on
        syield=syiel0
c      use this to minimize iterations during elastic deformation (
1)
c       deqpl=dtebs0*exp((smises-syield)*ratesf)
c      use this to minimize iterations during plastic deformation (
2)
        deqpl=esi
        do kewton=1,newton
          deqpl=max(deqpl,1.d-50)
          qhs=smises-eg3*deqpl-syield-rrates*dlog(deqpl/dtebs0)
          rhs=qhs+psf*shydro
          deqpl=deqpl+deqpl*rhs/(deqpl*(eg3+hard)+rrates)
          call uhard(syield,hard,eqplas+deqpl)
          if(abs(rhs).lt.toler*60.d0) goto 10
        end do
        write(7,2) newton
    2     format(//,30x,'***warning - plasticity algorithm did not
',
     1                    'converge after ',i3,' iterations')
        write(7,*)dstran(1),dstran(2),dstran(3),dstran(4)
        write(7,*)dstran(5),dstran(6),esi,smises,statev(1)
        write(7,*)statev(2),statev(3),statev(4),statev(5)
        write(7,*)qhs,deqpl,rhs,shydro,stress(1),stress(2)
        write(7,*)stress(3),stress(4),stress(5),stress(6)
   10   continue
c
c      the new equivalent deviatoric stress (q) is
        q=syield+rrates*dlog(deqpl/dtebs0)-psf*shydro
c
c      update stress, elastic and plastic strains and
c      equivalent plastic strain
        do k1=1,ndi
          stress(k1)=flow(k1)*q+shydro
        end do
        do k1=ndi+1,ntens
          stress(k1)=flow(k1)*q
        end do
        eqplas=eqplas+deqpl
c
```

FIG. 14C

source code implicit finite element solver

```
c       calculate plastic dissipation
          spd=deqpl*(qold+q)/two
c
c       formulate the jacobian (material tangent)
c       first calculate effective moduli
          effg=eg*q/smises
          effg2=two*effg
          effg3=three/two*effg2
          efflam=(ebulk3-effg2)/three
          hard1=hard+rrates/deqpl
          effhrd=eg3*hard1/(eg3+hard1)-effg3
          cee=-ebulk3*psf*eg*deqpl/smises
          do k1=1,ndi
            do k2=1,ndi
              ddsdde(k2,k1)=efflam+cee*flow(k2)
            end do
            ddsdde(k1,k1)=effg2+efflam+cee*flow(k1)
          end do
          do k1=ndi+1,ntens
            ddsdde(k1,k1)=effg
          end do
          do k1=1,ntens
            do k2=1,ntens
              ddsdde(k2,k1)=ddsdde(k2,k1)+effhrd*flow(k2)*flow(k1)
            end do
          end do
        endif
c
c     store state variables in array
c     equiv strain,mises stress,plastic strain rate,elastic strain
c     rate and iterations to convergence
        statev(1)=eqplas
        statev(2)=q
        statev(3)=deqpl/dtime
        statev(4)=esi/dtime
        statev(5)=kewton
c
        return
        end
c
        subroutine uhard(syield,hard,eqplas)
c
        include 'aba_param.inc'
c       table must be dimensioned correctly below:
        dimension table(2,7)
        parameter(zero=0.d0)
c       nbv 313 hardening table
          nvalue=7
c         this is room temp data
          table(1,1)=0.00d0
```

FIG. 14D

source code implicit finite element solver

```
      table(2,1)=0.0
      table(1,2)=-5.295d0
      table(2,2)=0.151
      table(1,3)=-3.04d0
      table(2,3)=0.337
      table(1,4)=4.726d0
      table(2,4)=0.542
      table(1,5)=14.41d0
      table(2,5)=0.736
      table(1,6)=48.146d0
      table(2,6)=1.093
      table(1,7)=2704.4d0
      table(2,7)=17.086
c
      do k1=1,nvalue-1
        eqpl1=table(2,k1+1)
        if(eqplas.lt.eqpl1) then
          eqpl0=table(2,k1)
c        current yield stress and hardening
c
          deqpl=eqpl1-eqpl0
          syiel0=table(1,k1)
          syiel1=table(1,k1+1)
          dsyiel=syiel1-syiel0
          hard=dsyiel/deqpl
          syield=syiel0+(eqplas-eqpl0)*hard
          goto 10
        endif
      end do
   10 continue
c
    return
    end
```

FIG. 14E

source code explicit finite element solver

```
c       vectorized user material subroutine for shell and plane
c       stress elements (abaqus5.5)
c       rate/temp dependent isotropic plasticity with linear
c       elasticity, strain softening/hardening & press. depnd.
c       yield
c       by omar a hasan (hasan@crd.ge.com)
c       last modified 05-03-96
c
        subroutine vumat(
c       read only variables (unmodifiable)
     1  nblock,ndir,nshr,nstatev,nfieldv,nprops,lanneal,
     2  step_time,total_time,dt,cmname,coord_mp,char_length,
     3  props,density,strain_inc,rel_spin_inc,
     4  temp_old,stretch_old,defgrad_old,field_old,
     5  stress_old,state_old,ener_intern_old,ener_inelas_old,
     6  temp_new,stretch_new,defgrad_new,field_new,
c       write only variables (modifiable)
     7  stress_new,state_new,ener_intern_new,ener_inelas_new)
c
        include 'vaba_param.inc'
c
        dimension coord_mp(nblock,*),char_length(nblock),props(npro
ps),
     1  density(nblock),strain_inc(nblock,ndir+nshr),
     2  rel_spin_inc(nblock,nshr),temp_old(nblock),
     3  stretch_old(nblock,ndir+nshr),
     4  defgrad_old(nblock,ndir+nshr+nshr),field_old(nblock,nfieldv
),
     5  stress_old(nblock,ndir+nshr),state_old(nblock,nstatev),
     6  ener_intern_old(nblock),ener_inelas_old(nblock),
     7  temp_new(nblock),stretch_new(nblock,ndir+nshr),
     8  defgrad_new(nblock,ndir+nshr+nshr),field_new(nblock,nfieldv
),
     9  stress_new(nblock,ndir+nshr),state_new(nblock,nstatev),
     1  ener_intern_new(nblock),ener_inelas_new(nblock)
c
        integer limit
        parameter (limit=40)
        dimension table(2,9)
        character*8 cmname
        parameter(zero=0.d0,one=1.d0,two=2.d0,three=3.d0,six=6.d0,
     1  four=4.d0,oneptf=1.5d0,zept=0.25d0,twbth=0.66666666666d0,
     2  eitee=80.d0)
c
c ---------------------------------------------------------------
c       props(1) - e- modulus (temperature dependent)
c       props(2) - nu- poisson ratio
c
c       Properties 3 and 4 descibe the rate sensitivity of yield ba
sed on a plot of
```

FIG 15A

source code explicit finite element solver

```
c       yield stress (x-axis) vs ln(strain rate) y-axis
c
c       props(3) - rate sensitivity (temperature dependent)   SLOPE
c       props(4) - intrinsic flow rate (temperature dependent)INTER
CPT
c
c       Property 5 descibes the pressure sensitivity of yield
c
c       props(5) - pressure sensitivity factor
c
c       Property 6 is the failure criterion ... either an equivalen
t plastic strain
c       for ductile failure or a maximum principal stress for britt
le failure
c
c       props(6) - failure criterion
c
c       NOTE -THESE FOLLOWING TWO LINES WOULD APPEAR IN THE ABAQUS
EXPLICIT
c               INPUT DECK
c
c       *USER MATERIAL,CONSTANTS=5
c       2.24e9,0.40,3.29e-7,1.48e-14,0.16
c       *DEPVAR,DELETE=6
c       8
c ------------------------------------------------------------------
c
c       material properties
        emod=props(1)
        enu=props(2)
        ebulk3=emod/(one-two*enu)
        eg2=emod/(one+enu)
        eg=eg2/two
        eg3=three*eg
        elam=(ebulk3-eg2)/three
        elp2g=elam+eg2
        ratesf=props(3)
        dtebs0=dt*props(4)
        psf=props(5)
        rrates=one/ratesf
        failst=props(6)
        table(1,1)=0.0
        table(2,1)=0.0
        table(1,2)=6.2
        table(2,2)=0.15
        table(1,3)=17.93
        table(2,3)=0.35
        table(1,4)=34.47
        table(2,4)=0.55
        table(1,5)=53.09
```

FIG 15B

source code explicit finite element solver

```
      table(2,5)=0.75
      table(1,6)=70.32
      table(2,6)=0.95
      table(1,7)=91.01
      table(2,7)=1.15
      table(1,8)=146.16
      table(2,8)=1.35
      table(1,9)=201.3
      table(2,9)=1.55
c
      do 100 i=1,nblock
c
c     initialize state variables
      eqplas=state_old(i,1)
      sm_old=state_old(i,2)
      icont=state_old(i,3)
      tstart=total_time-dt
      if (tstart.lt.1.e-6) then
      icont=1
      state_old(i,6)=one
      endif
c
      if (state_old(i,6).lt.0.5) then
      state_new(i,6)=zero
      goto 100
      endif
c
c     get hardening modulus and intrinsic resistance at t
      hard=(table(1,icont+1)-table(1,icont))/
     1    (table(2,icont+1)-table(2,icont))
      s_intr=table(1,icont)+hard*(eqplas-table(2,icont))
c
c     calculate predictor stress
      trace2=strain_inc(i,1)+strain_inc(i,2)
      del_e33=-elam*trace2/elp2g
      sig11o=stress_old(i,1)+eg2*strain_inc(i,1)
      sig22o=stress_old(i,2)+eg2*strain_inc(i,2)
      sig33=zero
      sig12=stress_old(i,4)+eg2*strain_inc(i,4)
      ss12s=six*(sig12**2)
c
c     since strain_inc(i,3) is not known apriori, loop 3
c     times without checking for convergence (works very well
c     in practise by reducing sig33 to 0.0000001*syield)
      do 200 ii=1,3
      trace=trace2+del_e33
      sig11=sig11o+elam*trace
      sig22=sig22o+elam*trace
c
c     calculate equivalent von mises stress from deviatoric
```

FIG. 15C

source code explicit finite element solver

```
c       component of trial (predictor) stress.
        smises=(sig11-sig22)**2+(sig22)**2+(sig11)**2
        smises=smises+ss12s
        smises=sqrt(smises/two)
c       avoid division by zero during first iteration
        smises=max(one,smises)
c
c       separate the hydrostatic from the deviatoric stress
c       calculate the flow direction
        shydro=(sig11+sig22)/three
        flow11=(sig11-shydro)/smises
        flow22=(sig22-shydro)/smises
        flow33=(sig33-shydro)/smises
        flow12=sig12/smises
c
c       solve for equivalent von mises stress and equivalent
c       plastic strain increment
        adfp=-psf*shydro*ratesf
        deqpl=dtebs0*exp((sm_old-s_intr)*ratesf+adfp)
        sm_new=smises-eg3*deqpl
c
c       update e33
        opfe=oneptf*deqpl
        d_ep11=opfe*flow11
        d_ep22=opfe*flow22
        d_ep33=opfe*flow33
        d_ep12=opfe*flow12
        d_ee11=strain_inc(i,1)-d_ep11
        d_ee22=strain_inc(i,2)-d_ep22
        d_ee33=-elam*(d_ee11+d_ee22)/elp2g
        d_ee12=strain_inc(i,4)-d_ep12
        del_e33=d_ee33+d_ep33
 200    continue
        esi=strain_inc(i,1)**2+strain_inc(i,2)**2+
     1  del_e33**2+two*strain_inc(i,4)**2
        esi=sqrt(esi*twbth)
        strain_inc(i,3)=del_e33
c
c       update stress, equivalent plastic strain, location
c       of plastic strain counter and state variables
        stress_new(i,1)=flow11*sm_new+shydro
        stress_new(i,2)=flow22*sm_new+shydro
        stress_new(i,3)=zero
        stress_new(i,4)=flow12*sm_new
        eqplas=eqplas+deqpl
        if (eqplas.gt.table(2,icont+1)) icont=icont+1
        cstate_new(i,1)=state_old(i,1)+d_ee11
        cstate_new(i,2)=state_old(i,2)+d_ee22
        cstate_new(i,3)=state_old(i,3)+d_ee12
        cstate_new(i,4)=state_old(i,4)+d_ep11
```

FIG. 15D

source code explicit finite element solver

```
          cstate_new(i,5)=state_old(i,5)+d_ep22
          cstate_new(i,6)=state_old(i,6)+d_ep12
c         save state variables: plastic strain, vm stress,, total
c         strain rate, plastic strain rate, failure criterion flag
          state_new(i,1)=eqplas
          state_new(i,2)=sm_new
          state_new(i,3)=icont
          state_new(i,4)=esi/dt
          state_new(i,5)=deqpl/dt
          state_new(i,6)=state_old(i,6)
c
          bee=-(stress_new(i,1)+stress_new(i,2))
          bee2=bee*bee
          cee=stress_new(i,1)*stress_new(i,2)-stress_new(i,4)*
     1    stress_new(i,4)
          froot=bee2-four*cee
          ffrot=max(one,froot)
          sqbm4c=sqrt(ffrot)
          pmax=(-bee+sqbm4c)/two
          pmin=(-bee-sqbm4c)/two
          state_new(i,7)=pmax
          state_new(i,8)=pmin
c         UNPAINTED
          failst=89.06
          if (pmax.gt.failst) state_new(i,6)=zero
          strain based failure criterion
          if (eqplas.gt.failst) state_new(i,6)=zero
c         update plastic dissipation
          plastic_work_inc=deqpl*(sm_old+sm_new)/two
          ener_inelas_new(i)=ener_inelas_old(i)+
     1    plastic_work_inc/density(i)
c
 100      continue
          return
          end
```

FIG 15E